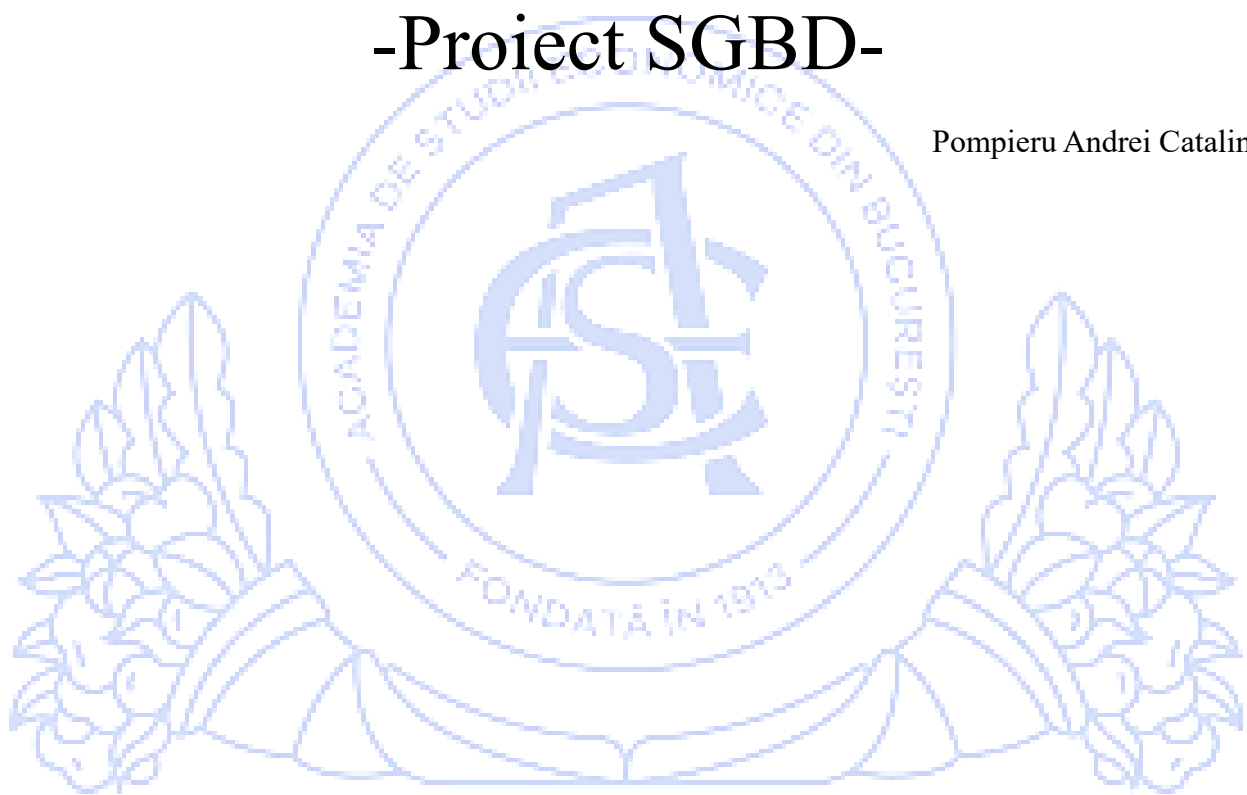


Gestionarea unui lant Hotelier

-Proiect SGBD-

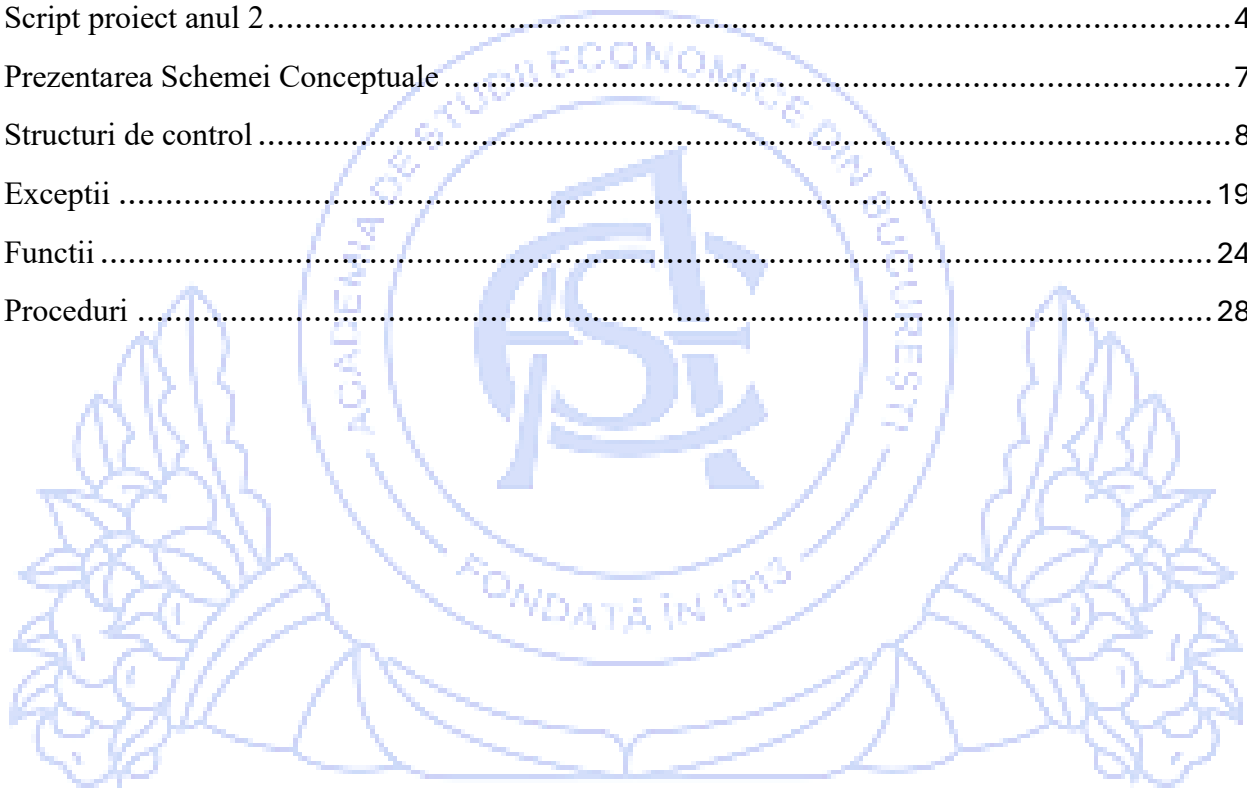
Pompieru Andrei Catalin



Cuprins

Contents

Cuprins	2
Descrierea Tematicii problemei	3
Script proiect anul 2	4
Prezentarea Schemei Conceptuale	7
Structuri de control	8
Exceptii	19
Functii	24
Proceduri	28



Descrierea Tematicii problemei

Proiectul vizează implementarea unui sistem de gestionare a rezervărilor în industria hotelieră. Scopul acestui sistem este să ofere o soluție eficientă pentru administrarea informațiilor legate de clienți, camere, rezervări și plăți în cadrul unei unități hoteliere. Sistemul va facilita monitorizarea și gestionarea rezervărilor, contribuind la optimizarea proceselor operaționale și îmbunătățirea experienței clienților.

Obiective principale:

- Administrarea eficientă a rezervărilor
- Gestionarea informațiilor despre clienți
- Monitorizarea tranzacțiilor financiare
- Îmbunătățirea experienței clienților

Funcționalitatea principală a sistemului

- Stocarea și gestionarea informațiilor despre clienți, inclusive date de contact, preferințe și istoric
- Monitorizarea disponibilității camerelor, prețurilor și facilităților
- Gestionarea rezervărilor active, istoricul rezervărilor și statuturile acestora (confirmate, anulate etc.)
- Înregistrarea tranzacțiilor și asocierea acestora cu rezervările aferente
- Colectarea feedback-ului clienților pentru îmbunătățirea serviciilor și generarea de rapoarte utile .

Script proiect anul 2

-- Tabela pentru clienți

```
CREATE TABLE clienti (
  ID_CLIENT INT PRIMARY KEY,
  nume VARCHAR(50) NOT NULL,
  prenume VARCHAR(50) NOT NULL,
  adresa VARCHAR(100),
  email VARCHAR(50),
  data_inregistrare DATE,
  nr_telefon VARCHAR(15) );
```

-- Tabela pentru camere

```
CREATE TABLE camere (
  ID_CAMERA INT PRIMARY KEY,
  numar_camera INT,
  pret_noapte DECIMAL(8, 2) NOT NULL,
  facilitati VARCHAR(50) );
```

-- Tabela pentru rezervări

```
CREATE TABLE rezervari (
  id_rezervare INT PRIMARY KEY,
  id_client INT,
  id_camera INT,
  data_check_in DATE,
  data_check_out DATE,
  stare_rezervare VARCHAR(20),
  FOREIGN KEY (id_client) REFERENCES clienti(ID_CLIENT),
  FOREIGN KEY (id_camera) REFERENCES camere(ID_CAMERA) );
```

-- Tabela pentru plăți

```
CREATE TABLE plati (
  id_plata INT PRIMARY KEY,
  id_rezervare INT,
  data_plata DATE,
  suma_plata DECIMAL(8, 2),
  id_client INT,
  FOREIGN KEY (id_rezervare) REFERENCES rezervari(id_rezervare),
  FOREIGN KEY (id_client) REFERENCES clienti(ID_CLIENT) );
```

-- Tabela pentru feedback

```
CREATE TABLE feedback (
  ID_FEEDBACK INT PRIMARY KEY,
  id_client INT,
  mesaj VARCHAR(100),
  data_feedback DATE,
  FOREIGN KEY (ID_client) REFERENCES clienti(ID_CLIENT) );
```

--FOLOSIM COMANDA DROP

```
DROP TABLE feedback;
```

--FOLOSIM ALTER

```
ALTER TABLE CAMERE RENAME TO CAMERE_HOTEL;
```

--ALTER TABLE rezervari

```
ALTER TABLE rezervari
```

```
RENAME TO istoric_rezervari;
```

```

INSERT INTO clienti (ID_CLIENT, nume, prenume, adresa, email, data_inregistrare, nr_telefon)
VALUES (1, 'Popescu', 'Ion', 'Str. Principala 10', 'popescu@gmail.com', TO_DATE('2023-01-01', 'YYYY-MM-DD'),
'0723456789');

INSERT INTO clienti (ID_CLIENT, nume, prenume, adresa, email, data_inregistrare, nr_telefon)
VALUES (2, 'David', 'Ana', 'Str. Mare 2', 'davidanda@gmail.com', TO_DATE('2023-02-22', 'YYYY-MM-DD'), '0723421313');

INSERT INTO clienti (ID_CLIENT, nume, prenume, adresa, email, data_inregistrare, nr_telefon)
VALUES (3, 'Anca', 'Andra', 'Str. Mica 3', 'ancaandra@gmail.com', TO_DATE('2022-03-22', 'YYYY-MM-DD'),
'0723213113');

INSERT INTO clienti (ID_CLIENT, nume, prenume, adresa, email, data_inregistrare, nr_telefon)
VALUES (4, 'Pompieru', 'Andrei Cătălin', 'Str. Uriasa 29', 'pompieruandrei@gmail.com', TO_DATE('2003-05-29', 'YYYY-MM-DD'), '0713236789');

INSERT INTO clienti (ID_CLIENT, nume, prenume, adresa, email, data_inregistrare, nr_telefon)
VALUES (5, 'Radu', 'Elena', 'Str. Soarelui 7', 'elena.radu@example.com', TO_DATE('2023-04-05', 'YYYY-MM-DD'),
'0721345678');

INSERT INTO clienti (ID_CLIENT, nume, prenume, adresa, email, data_inregistrare, nr_telefon)
VALUES (6, 'Dumitru', 'Andrei', 'Bd. Unirii 14', 'andrei.dumitru@example.com', TO_DATE('2023-05-20', 'YYYY-MM-DD'),
'0755432112');

INSERT INTO clienti (ID_CLIENT, nume, prenume, adresa, email, data_inregistrare, nr_telefon)
VALUES (7, 'Gheorghe', 'Cristina', 'Str. Izvorul Muresului 3', 'cristina.gheorghe@example.com', TO_DATE('2023-06-10', 'YYYY-MM-DD'), '0766001122');

INSERT INTO clienti (ID_CLIENT, nume, prenume, adresa, email, data_inregistrare, nr_telefon)
VALUES (8, 'Munteanu', 'George', 'Bd. Decebal 19', 'george.munteanu@example.com', TO_DATE('2023-07-15', 'YYYY-MM-DD'), '0722121212');

INSERT INTO clienti (ID_CLIENT, nume, prenume, adresa, email, data_inregistrare, nr_telefon)
VALUES (9, 'Diaconu', 'Elena', 'Str. Alba Iulia 8', 'elena.diaconu@example.com', TO_DATE('2023-08-02', 'YYYY-MM-DD'), '0733223344');

INSERT INTO clienti (ID_CLIENT, nume, prenume, adresa, email, data_inregistrare, nr_telefon)
VALUES (10, 'Constantin', 'Andreea', 'Bd. Iancu de Hunedoara 12', 'andreea.constantin@example.com', TO_DATE('2023-09-18', 'YYYY-MM-DD'), '0711567890');

```

```
INSERT ALL
```

```
    INTO camere (ID_CAMERA, numar_camera, pret_noapte, facilitati)
```

```
VALUES (101, 1, 100.00, 'TV, WiFi')
```

```
    INTO camere (ID_CAMERA, numar_camera, pret_noapte, facilitati)
```

```
VALUES (102, 2, 150.00, 'TV, WiFi, Aer')
```

```
    INTO camere (ID_CAMERA, numar_camera, pret_noapte, facilitati)
```

```
VALUES (103, 3, 120.00, 'TV, WiFi, Balcon')
```

```
    INTO camere (ID_CAMERA, numar_camera, pret_noapte, facilitati)
```

```
VALUES (104, 4, 80.00, 'TV, WiFi')
```

```
    INTO camere (ID_CAMERA, numar_camera, pret_noapte, facilitati)
```

```
VALUES (105, 5, 200.00, 'TV, WiFi, Aer, Jacuzzi')
```

```
    INTO camere (ID_CAMERA, numar_camera, pret_noapte, facilitati)
```

```
VALUES (106, 6, 90.00, 'TV, WiFi, Balcon')
```

```
    INTO camere (ID_CAMERA, numar_camera, pret_noapte, facilitati)
```

```
VALUES (107, 7, 130.00, 'TV, WiFi, Aer')
```

```
    INTO camere (ID_CAMERA, numar_camera, pret_noapte, facilitati)
```

```
VALUES (108, 8, 110.00, 'TV, WiFi')
```

```
    INTO camere (ID_CAMERA, numar_camera, pret_noapte, facilitati)
```

```
VALUES (109, 9, 75.00, 'TV, WiFi')
```

```
    INTO camere (ID_CAMERA, numar_camera, pret_noapte, facilitati)
```

```
VALUES (110, 10, 180.00, 'TV, WiFi, Aer')
```

```
SELECT 1 FROM DUAL;
```

```
INSERT INTO istoric_rezervari (id_rezervare, id_client, id_camera, data_check_in, data_check_out, stare_rezervare)
```

```
VALUES (1, 1, 101, TO_DATE('2023-03-15', 'YYYY-MM-DD'), TO_DATE('2023-03-20', 'YYYY-MM-DD'), 'Activ');
```

```
INSERT INTO istoric_rezervari (id_rezervare, id_client, id_camera, data_check_in, data_check_out, stare_rezervare)
```

```
VALUES (2, 2, 102, TO_DATE('2023-04-10', 'YYYY-MM-DD'), TO_DATE('2023-04-15', 'YYYY-MM-DD'), 'Inactiv');
```

```
INSERT INTO istoric_rezervari (id_rezervare, id_client, id_camera, data_check_in, data_check_out, stare_rezervare)
```

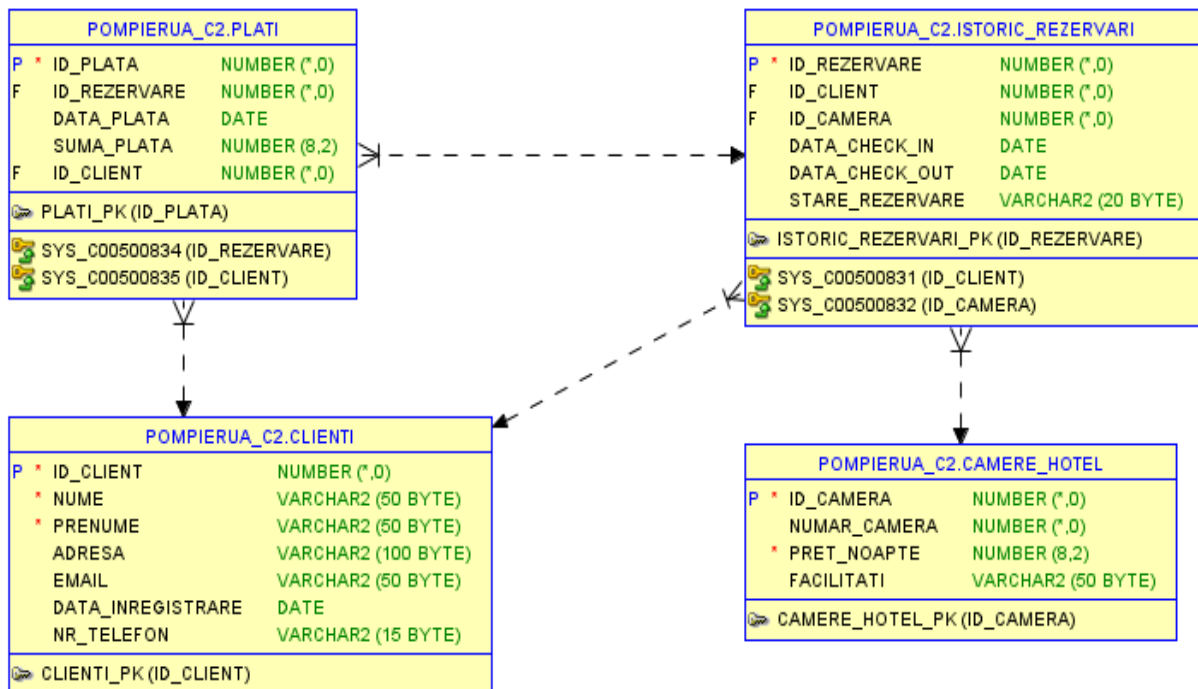
```
VALUES (3, 3, 103, TO_DATE('2023-05-01', 'YYYY-MM-DD'), TO_DATE('2023-05-05', 'YYYY-MM-DD'), 'Activ');
```

```
INSERT INTO istoric_rezervari (id_rezervare, id_client, id_camera, data_check_in, data_check_out, stare_rezervare)
```

```
INSERT INTO plati (id_plata, id_rezervare, data_plata, suma_plata, id_client)
```

```
VALUES (1, 1, TO_DATE('2023-03-18', 'YYYY-MM-DD'), 500.00, 1);
```

Prezentarea Schemei Conceptuale



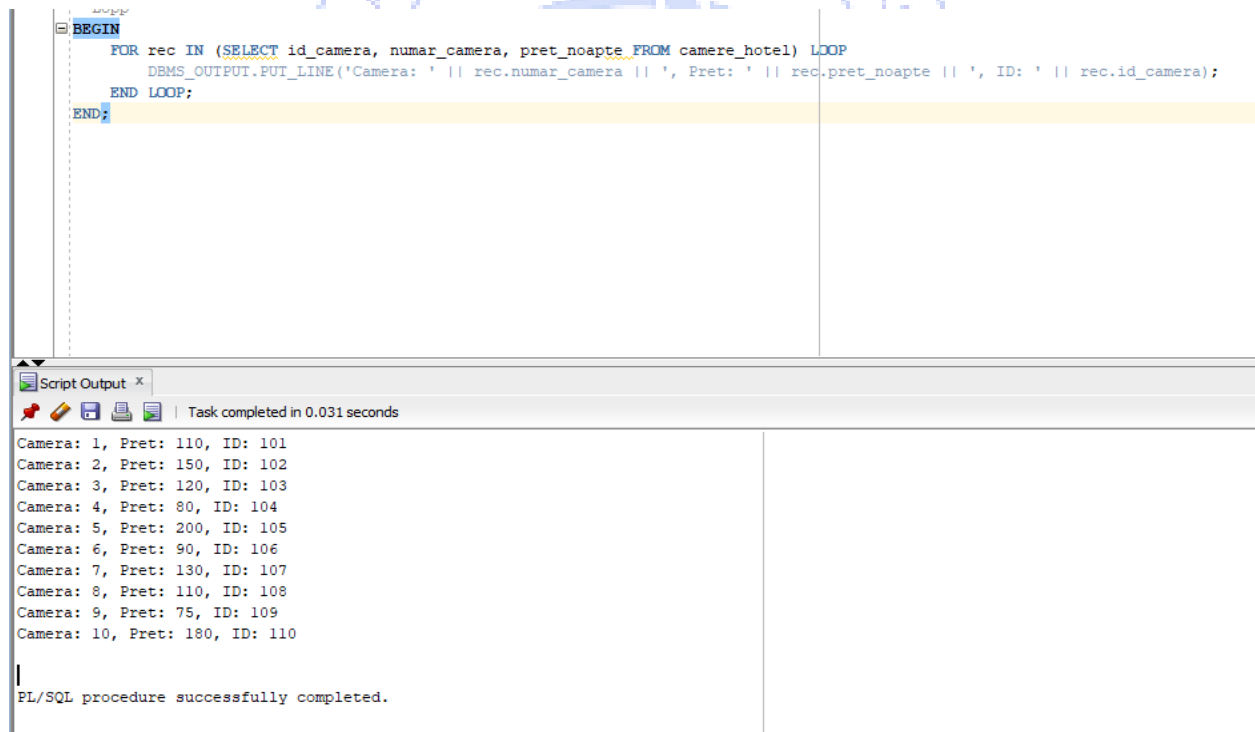
Structuri de control

--FOR LOOP

```
BEGIN
```

```
    FOR rec IN (SELECT id_camera, numar_camera, pret_noapte FROM camere_hotel) LOOP
        DBMS_OUTPUT.PUT_LINE('Camera: ' || rec.numar_camera || ', Pret: ' || rec.pret_noapte
|| ', ID: ' || rec.id_camera);
    END LOOP;
```

```
END;
```



The screenshot shows a PL/SQL script editor with the following code:

```
BEGIN
    FOR rec IN (SELECT id_camera, numar_camera, pret_noapte FROM camere_hotel) LOOP
        DBMS_OUTPUT.PUT_LINE('Camera: ' || rec.numar_camera || ', Pret: ' || rec.pret_noapte || ', ID: ' || rec.id_camera);
    END LOOP;
END;
```

Below the editor, the 'Script Output' window displays the results of the execution:

```
Task completed in 0.031 seconds

Camera: 1, Pret: 110, ID: 101
Camera: 2, Pret: 150, ID: 102
Camera: 3, Pret: 120, ID: 103
Camera: 4, Pret: 80, ID: 104
Camera: 5, Pret: 200, ID: 105
Camera: 6, Pret: 90, ID: 106
Camera: 7, Pret: 130, ID: 107
Camera: 8, Pret: 110, ID: 108
Camera: 9, Pret: 75, ID: 109
Camera: 10, Pret: 180, ID: 110

PL/SQL procedure successfully completed.
```

--WHILE LOOP

```
DECLARE
```

```
    v_count NUMBER := 1;
```



```
BEGIN
```

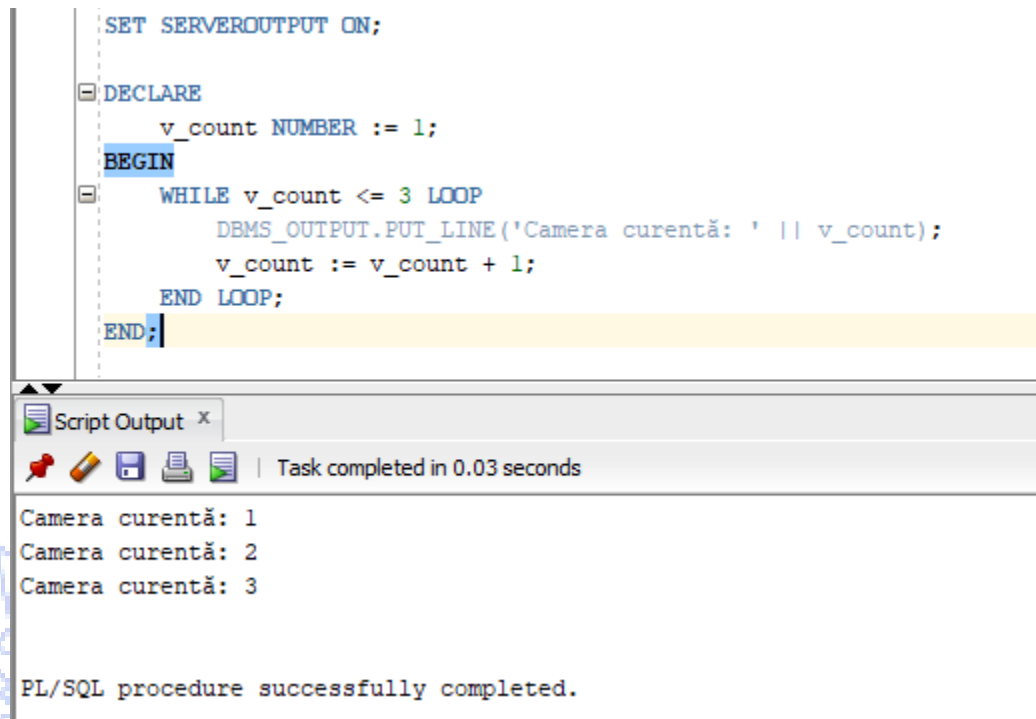
```
    WHILE v_count <= 3 LOOP
```

```
        DBMS_OUTPUT.PUT_LINE('Camera curentă: ' || v_count);
```

```
        v_count := v_count + 1;
```

```
    END LOOP;
```

```
END;
```



```
SET SERVEROUTPUT ON;

DECLARE
    v_count NUMBER := 1;
BEGIN
    WHILE v_count <= 3 LOOP
        DBMS_OUTPUT.PUT_LINE('Camera curentă: ' || v_count);
        v_count := v_count + 1;
    END LOOP;
END;
```

Script Output x

Task completed in 0.03 seconds

```
Camera curentă: 1
Camera curentă: 2
Camera curentă: 3

PL/SQL procedure successfully completed.
```

-IF-THEN-ELSE

```
DECLARE
```

```
    v_total_plati NUMBER(10, 2);
```

```
    v_id_client NUMBER := 10;
```

```
BEGIN
```

```
    SELECT SUM(suma_plata)
```

```
    INTO v_total_plati
```

```
    FROM plati
```

```
    WHERE id_client = v_id_client;
```

```
IF v_total_plati IS NULL THEN
    DBMS_OUTPUT.PUT_LINE('Clientul nu a efectuat nicio plata.');
```

ELSIF v_total_plati < 500 THEN

```
    DBMS_OUTPUT.PUT_LINE('Clientul a efectuat plati mici: ' || v_total_plati);
```

ELSIF v_total_plati BETWEEN 500 AND 2000 THEN

```
    DBMS_OUTPUT.PUT_LINE('Clientul a efectuat plati medii: ' || v_total_plati);
```

ELSE

```
    DBMS_OUTPUT.PUT_LINE('Clientul a efectuat plati mari: ' || v_total_plati);
```

END IF;

END;

--Vom testa programul pentru 2 clienti in SS se va observa ca prima oara a fost rulat pentru clientul cu ID-ul 1



```

DECLARE
    v_total_plati NUMBER(10, 2);
    v_id_client NUMBER := 10;
BEGIN

    SELECT SUM(suma_plata)
    INTO v_total_plati
    FROM plati
    WHERE id_client = v_id_client;

    IF v_total_plati IS NULL THEN
        DBMS_OUTPUT.PUT_LINE('Clientul nu a efectuat nicio plata.');
```

```

    ELSIF v_total_plati < 500 THEN
        DBMS_OUTPUT.PUT_LINE('Clientul a efectuat plati mici: ' || v_total_plati);
    ELSIF v_total_plati BETWEEN 500 AND 2000 THEN
        DBMS_OUTPUT.PUT_LINE('Clientul a efectuat plati medii: ' || v_total_plati);
    ELSE
        DBMS_OUTPUT.PUT_LINE('Clientul a efectuat plati mari: ' || v_total_plati);
    END IF;
END;
```

Script Output x

Task completed in 0.025 seconds

Camera curenta: 3

PL/SQL procedure successfully completed.

Clientul a efectuat plati medii: 500

PL/SQL procedure successfully completed.

Clientul a efectuat plati medii: 700

DECLARE

v_id_client NUMBER := 777;

v_nume_client VARCHAR2(50);

BEGIN

BEGIN

SELECT nume INTO v_nume_client

FROM clienti

WHERE id_client = v_id_client;

DBMS_OUTPUT.PUT_LINE('Numele clientului este: ' || v_nume_client);

EXCEPTION

```
WHEN NO_DATA_FOUND THEN
```

```
    DBMS_OUTPUT.PUT_LINE('Nu exista clientul cu acest ID.');
```

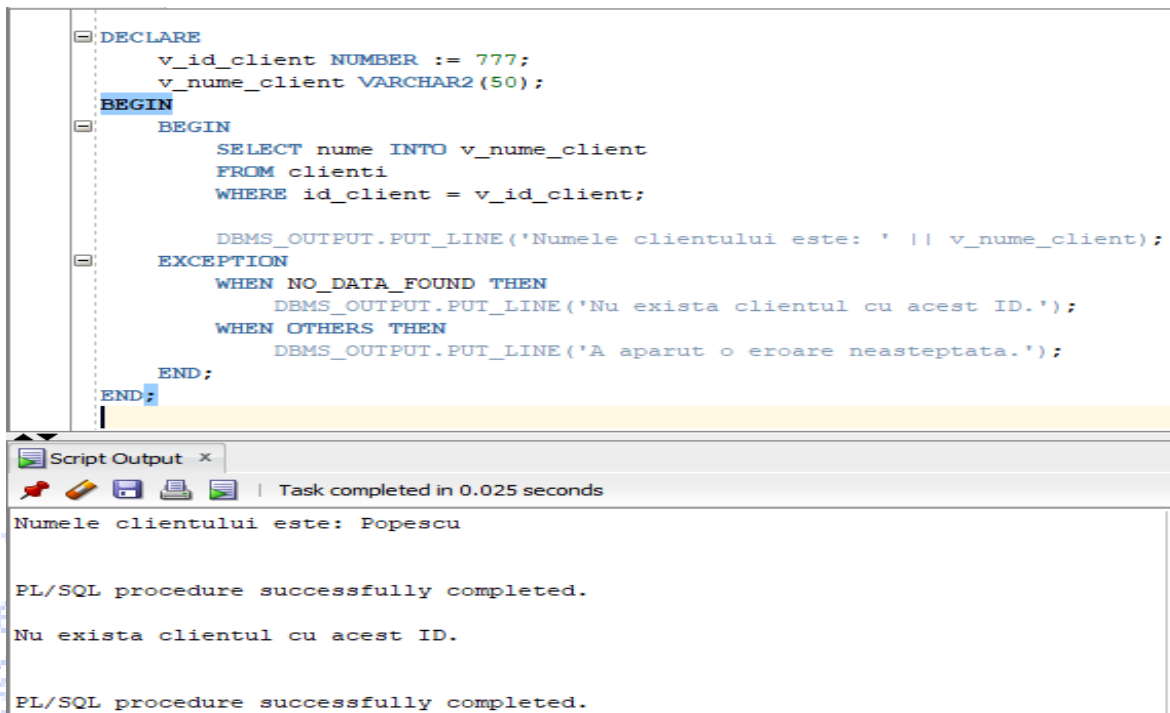
```
WHEN OTHERS THEN
```

```
    DBMS_OUTPUT.PUT_LINE('A aparut o eroare neasteptata.');
```

```
END;
```

```
END;
```

```
--vom testa mai intai un id existent, iar apoi unul care nu exista
```



```
DECLARE
    v_id_client NUMBER := 777;
    v_nume_client VARCHAR2(50);
BEGIN
    BEGIN
        SELECT nume INTO v_nume_client
        FROM clienti
        WHERE id_client = v_id_client;

        DBMS_OUTPUT.PUT_LINE('Numele clientului este: ' || v_nume_client);
    EXCEPTION
        WHEN NO_DATA_FOUND THEN
            DBMS_OUTPUT.PUT_LINE('Nu exista clientul cu acest ID.');
```

Script Output x

Task completed in 0.025 seconds

Numele clientului este: Popescu

PL/SQL procedure successfully completed.

Nu exista clientul cu acest ID.

PL/SQL procedure successfully completed.

Utilizarea Cursorilor

--CURSOR IMPLICIT

DECLARE

v_num clienti.num%TYPE;

v_prename clienti.prenume%TYPE;

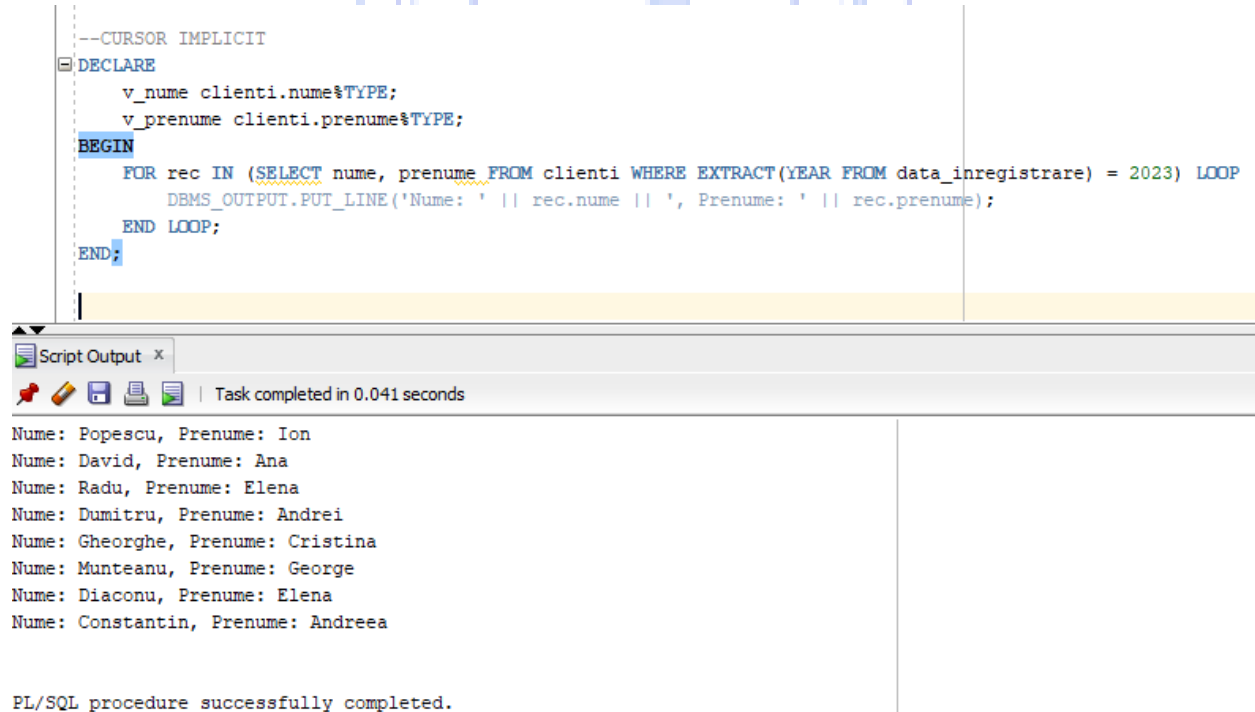
BEGIN

FOR rec IN (SELECT nume, prenume FROM clienti WHERE EXTRACT(YEAR FROM data_inregistrare) = 2023) LOOP

DBMS_OUTPUT.PUT_LINE('Nume: ' || rec.nume || ', Prenume: ' || rec.prenume);

END LOOP;

END;



```
--CURSOR IMPLICIT
DECLARE
    v_num clienti.num%TYPE;
    v_prename clienti.prenume%TYPE;
BEGIN
    FOR rec IN (SELECT nume, prenume FROM clienti WHERE EXTRACT(YEAR FROM data_inregistrare) = 2023) LOOP
        DBMS_OUTPUT.PUT_LINE('Nume: ' || rec.nume || ', Prenume: ' || rec.prenume);
    END LOOP;
END;
```

Script Output x

Task completed in 0.041 seconds

Nume: Popescu, Prenume: Ion
 Nume: David, Prenume: Ana
 Nume: Radu, Prenume: Elena
 Nume: Dumitru, Prenume: Andrei
 Nume: Gheorghe, Prenume: Cristina
 Nume: Munteanu, Prenume: George
 Nume: Diaconu, Prenume: Elena
 Nume: Constantin, Prenume: Andreea

PL/SQL procedure successfully completed.

--CURSOR EXPLICIT

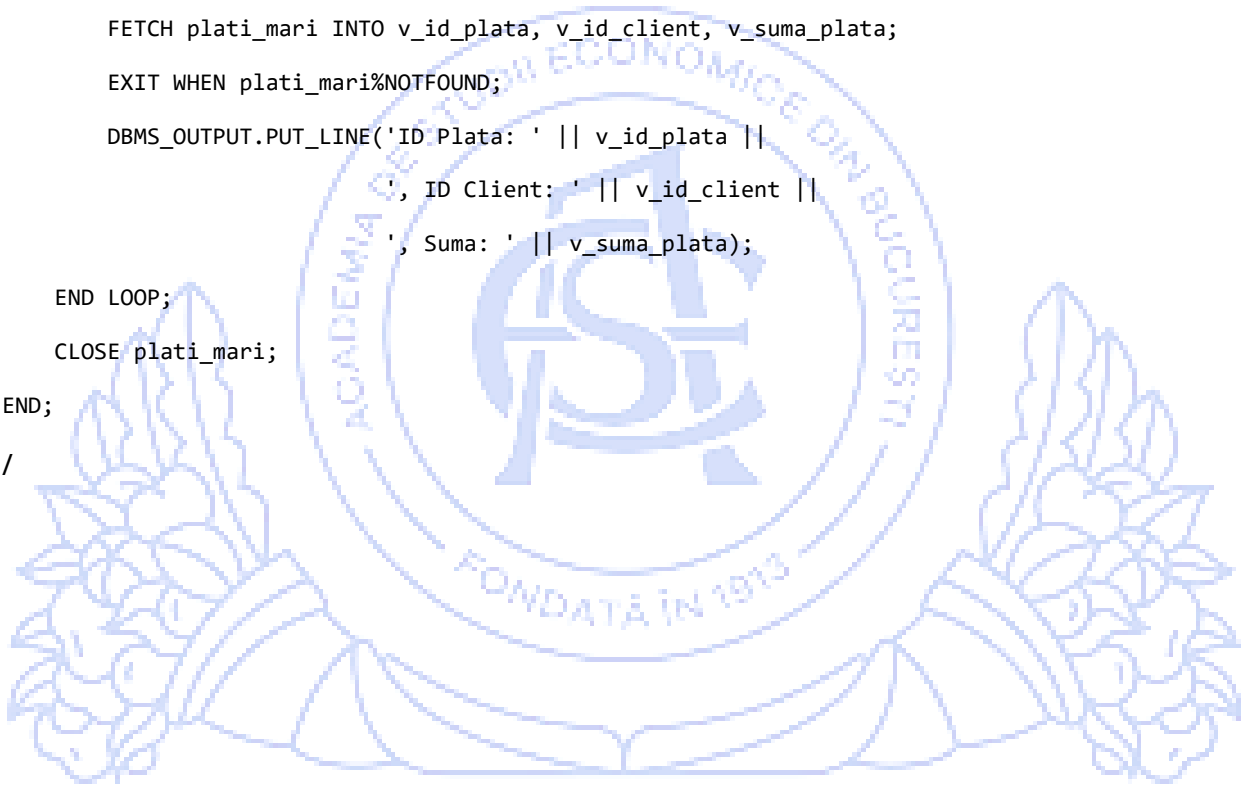
DECLARE

CURSOR plati_mari IS

```
SELECT id_plata, id_client, suma_plata
FROM plati
WHERE suma_plata > 600;

v_id_plata plati.id_plata%TYPE;
v_id_client plati.id_client%TYPE;
v_suma_plata plati.suma_plata%TYPE;

BEGIN
  OPEN plati_mari;
  LOOP
    FETCH plati_mari INTO v_id_plata, v_id_client, v_suma_plata;
    EXIT WHEN plati_mari%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE('ID Plata: ' || v_id_plata ||
                          ', ID Client: ' || v_id_client ||
                          ', Suma: ' || v_suma_plata);
  END LOOP;
  CLOSE plati_mari;
END;
/
```



```

--CURSOR EXPLICIT
DECLARE
    CURSOR plati_mari IS
        SELECT id_plata, id_client, suma_plata
        FROM plati
        WHERE suma_plata > 600;

    v_id_plata plati.id_plata%TYPE;
    v_id_client plati.id_client%TYPE;
    v_suma_plata plati.suma_plata%TYPE;
BEGIN
    OPEN plati_mari;
    LOOP
        FETCH plati_mari INTO v_id_plata, v_id_client, v_suma_plata;
        EXIT WHEN plati_mari%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE('ID Plata: ' || v_id_plata ||
                               ', ID Client: ' || v_id_client ||
                               ', Suma: ' || v_suma_plata);

    END LOOP;
    CLOSE plati_mari;
END;
/

```

Script Output x
Task completed in 0.028 seconds

```

ID Plata: 2, ID Client: 2, Suma: 750
ID Plata: 4, ID Client: 4, Suma: 900
ID Plata: 6, ID Client: 6, Suma: 1100
ID Plata: 7, ID Client: 7, Suma: 680
ID Plata: 8, ID Client: 8, Suma: 820
ID Plata: 10, ID Client: 10, Suma: 700

```

PL/SQL procedure successfully completed.

--CURSOR FOR LOOP

```

DECLARE
    v_id_rezervare istoric_rezervari.id_rezervare%TYPE;
    v_id_client istoric_rezervari.id_client%TYPE;
    v_id_camera istoric_rezervari.id_camera%TYPE;
    v_data_check_in istoric_rezervari.data_check_in%TYPE;
    v_data_check_out istoric_rezervari.data_check_out%TYPE;
BEGIN
    FOR rezervare IN (
        SELECT id_rezervare, id_client, id_camera, data_check_in, data_check_out
        FROM istoric_rezervari
        WHERE stare_rezervare = 'Activ'
    ) LOOP
        -- Stocăm valorile din fiecare rând în variabile
        v_id_rezervare := rezervare.id_rezervare;
        v_id_client := rezervare.id_client;
        v_id_camera := rezervare.id_camera;

```

```

v_data_check_in := rezervare.data_check_in;

v_data_check_out := rezervare.data_check_out;

-- Afişăm informațiile rezervării

DBMS_OUTPUT.PUT_LINE('ID Rezervare: ' || v_id_rezervare ||

                      ', ID Client: ' || v_id_client ||

                      ', ID Camera: ' || v_id_camera ||

                      ', Check-in: ' || v_data_check_in ||


                      ', Check-out: ' || v_data_check_out);

END LOOP;

END;

/

```



```

DECLARE
v_id_rezervare istoric_rezervari.id_rezervare%TYPE;
v_id_client istoric_rezervari.id_client%TYPE;
v_id_camera istoric_rezervari.id_camera%TYPE;
v_data_check_in istoric_rezervari.data_check_in%TYPE;
v_data_check_out istoric_rezervari.data_check_out%TYPE;
BEGIN
FOR rezervare IN (
SELECT id_rezervare, id_client, id_camera, data_check_in, data_check_out
FROM istoric_rezervari
WHERE stare_rezervare = 'Activ'
) LOOP
-- Stocăm valorile din fiecare rând în variabile
v_id_rezervare := rezervare.id_rezervare;
v_id_client := rezervare.id_client;
v_id_camera := rezervare.id_camera;
v_data_check_in := rezervare.data_check_in;
v_data_check_out := rezervare.data_check_out;

-- Afişăm informațiile rezervării
DBMS_OUTPUT.PUT_LINE('ID Rezervare: ' || v_id_rezervare ||

                      ', ID Client: ' || v_id_client ||

                      ', ID Camera: ' || v_id_camera ||

                      ', Check-in: ' || v_data_check_in ||

                      ', Check-out: ' || v_data_check_out);

END LOOP;
END;

```

Script Output x

Task completed in 0.039 seconds

ID Rezervare: 1,	ID Client: 1,	ID Camera: 101,	Check-in: 15-MAR-23,	Check-out: 20-MAR-23
ID Rezervare: 3,	ID Client: 3,	ID Camera: 103,	Check-in: 01-MAY-23,	Check-out: 05-MAY-23
ID Rezervare: 5,	ID Client: 5,	ID Camera: 105,	Check-in: 10-JUL-23,	Check-out: 15-JUL-23
ID Rezervare: 7,	ID Client: 7,	ID Camera: 107,	Check-in: 01-SEP-23,	Check-out: 05-SEP-23
ID Rezervare: 9,	ID Client: 9,	ID Camera: 109,	Check-in: 01-NOV-23,	Check-out: 05-NOV-23

--CURSOR EXPLICIT


```
DECLARE

CURSOR total_plati_client IS

    SELECT id_client, SUM(suma_plata) AS total

    FROM plati

    GROUP BY id_client;

v_id_client plati.id_client%TYPE;

v_total NUMBER;

BEGIN

    OPEN total_plati_client;

    LOOP

        FETCH total_plati_client INTO v_id_client, v_total;

        EXIT WHEN total_plati_client%NOTFOUND;

        DBMS_OUTPUT.PUT_LINE('ID Client: ' || v_id_client || ', Total plati: ' || v_total);

    END LOOP;

    CLOSE total_plati_client;

END;
```



```
--CURSOR EXPLICIT
DECLARE
    CURSOR total_plati_client IS
        SELECT id_client, SUM(suma_plata) AS total
        FROM plati
        GROUP BY id_client;
    v_id_client plati.id_client%TYPE;
    v_total NUMBER;
BEGIN
    OPEN total_plati_client;
    LOOP
        FETCH total_plati_client INTO v_id_client, v_total;
        EXIT WHEN total_plati_client%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE('ID Client: ' || v_id_client || ', Total plati: ' || v_total);
    END LOOP;
    CLOSE total_plati_client;
END;
```

Script Output x

Task completed in 0.031 seconds

```
ID Client: 1, Total plati: 500
ID Client: 2, Total plati: 750
ID Client: 4, Total plati: 900
ID Client: 5, Total plati: 450
ID Client: 6, Total plati: 1100
ID Client: 7, Total plati: 680
ID Client: 8, Total plati: 820
ID Client: 9, Total plati: 550
ID Client: 10, Total plati: 700
```

'L/SQL procedure successfully completed.



Exceptii

--Folosim exceptia NOT DATA FOUND sa cautam un client daca exista sau nu

DECLARE

```
v_id_rezervare ISTORIC_REZERVARI.ID_REZERVARE%TYPE;
v_id_client ISTORIC_REZERVARI.ID_CLIENT%TYPE;
v_id_camera ISTORIC_REZERVARI.ID_CAMERA%TYPE;
v_data_check_in ISTORIC_REZERVARI.DATA_CHECK_IN%TYPE;
v_data_check_out ISTORIC_REZERVARI.DATA_CHECK_OUT%TYPE;
```

BEGIN

```
v_id_rezervare := &Introduce_ID_REZERVARE;
```

```
SELECT ID_CLIENT, ID_CAMERA, DATA_CHECK_IN, DATA_CHECK_OUT
INTO v_id_client, v_id_camera, v_data_check_in, v_data_check_out
FROM ISTORIC_REZERVARI
WHERE ID_REZERVARE = v_id_rezervare;
```

```
DBMS_OUTPUT.PUT_LINE('Rezervare găsită: ID Client: ' || v_id_client ||
    ', ID Camera: ' || v_id_camera ||
    ', Check-in: ' || v_data_check_in ||
    ', Check-out: ' || v_data_check_out);
```

EXCEPTION

```
WHEN NO_DATA_FOUND THEN
```

```
    DBMS_OUTPUT.PUT_LINE('Eroare: Rezervarea cu ID-ul ' || v_id_rezervare || ' nu
    există.');
```

END;

/DECLARE

```
v_id_client CLIENTI.ID_CLIENT%TYPE;
```

```

v_num_client CLIENTI.NUM_CLIENT%TYPE;

v_prename_client CLIENTI.PRENAME_CLIENT%TYPE;

BEGIN

v_id_client := &Introduce_ID_CLIENT;

SELECT NUM_CLIENT, PRENAME_CLIENT
INTO v_num_client, v_prename_client
FROM CLIENTI
WHERE ID_CLIENT = v_id_client;

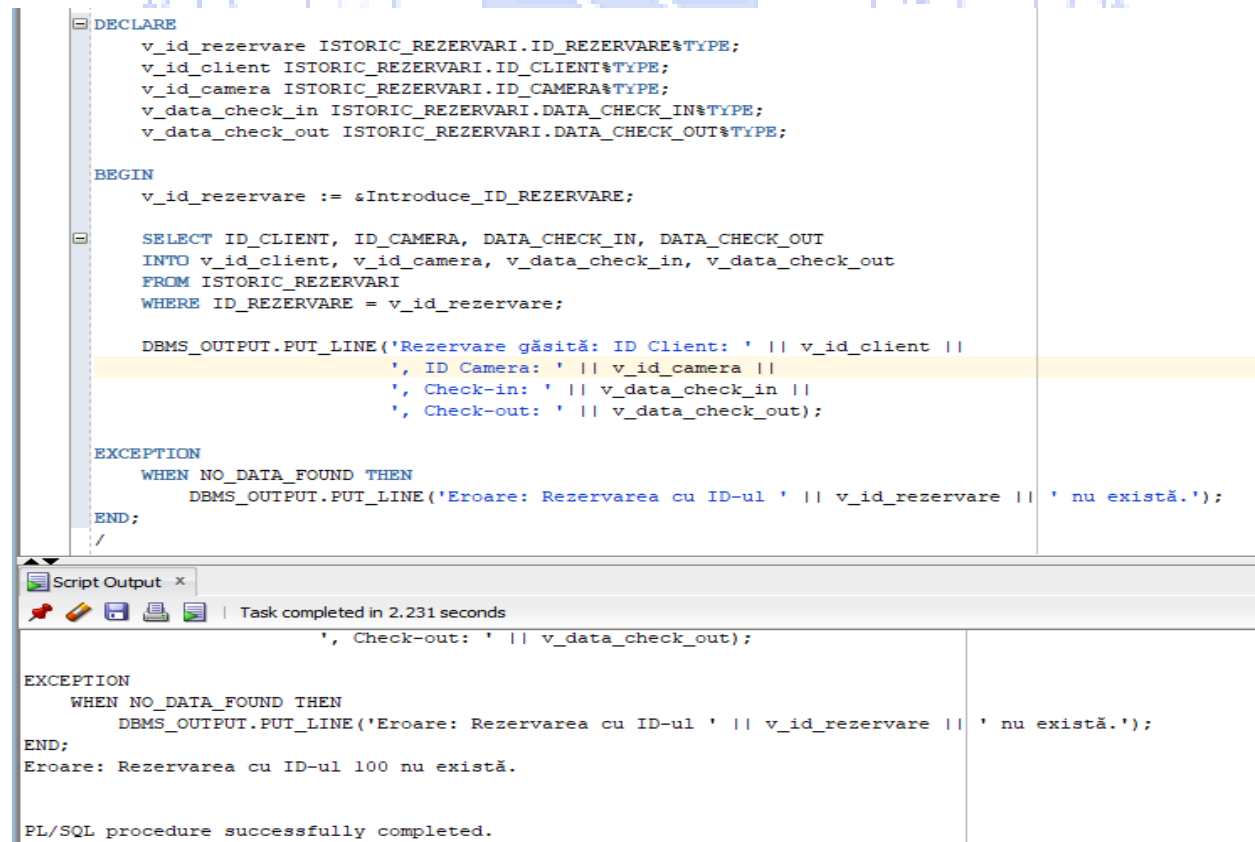
DBMS_OUTPUT.PUT_LINE('Client gasit: ' || v_num_client || ' ' || v_prename_client);

EXCEPTION

WHEN NO_DATA_FOUND THEN

DBMS_OUTPUT.PUT_LINE('Eroare: Clientul cu ID-ul ' || v_id_client || ' nu exista.');
```

END;



```

DECLARE
v_id_rezervare ISTORIC_REZERVARI.ID_REZERVARE%TYPE;
v_id_client ISTORIC_REZERVARI.ID_CLIENT%TYPE;
v_id_camera ISTORIC_REZERVARI.ID_CAMERA%TYPE;
v_data_check_in ISTORIC_REZERVARI.DATA_CHECK_IN%TYPE;
v_data_check_out ISTORIC_REZERVARI.DATA_CHECK_OUT%TYPE;

BEGIN
v_id_rezervare := &Introduce_ID_REZERVARE;

SELECT ID_CLIENT, ID_CAMERA, DATA_CHECK_IN, DATA_CHECK_OUT
INTO v_id_client, v_id_camera, v_data_check_in, v_data_check_out
FROM ISTORIC_REZERVARI
WHERE ID_REZERVARE = v_id_rezervare;

DBMS_OUTPUT.PUT_LINE('Rezervare găsită: ID Client: ' || v_id_client ||
', ID Camera: ' || v_id_camera ||
', Check-in: ' || v_data_check_in ||
', Check-out: ' || v_data_check_out);

EXCEPTION
WHEN NO_DATA_FOUND THEN
DBMS_OUTPUT.PUT_LINE('Eroare: Rezervarea cu ID-ul ' || v_id_rezervare || ' nu există.');
```

END;

/

Script Output x

Task completed in 2.231 seconds

```

', Check-out: ' || v_data_check_out);

EXCEPTION
WHEN NO_DATA_FOUND THEN
DBMS_OUTPUT.PUT_LINE('Eroare: Rezervarea cu ID-ul ' || v_id_rezervare || ' nu există.');
```

END;

Eroare: Rezervarea cu ID-ul 100 nu există.

PL/SQL procedure successfully completed.

--Exceptie DUP_VAL_ON_INDEX pentru inserarea unei camera duplicate

```
BEGIN
```

```
    INSERT INTO CAMERE_HOTEL (ID_CAMERA, NUMAR_CAMERA, PRET_NOAPTE, FACILITATI)
    VALUES (101, 1, 100.00, 'TV, WiFi');
```

```
    DBMS_OUTPUT.PUT_LINE('Camera adaugata cu succes.');
```

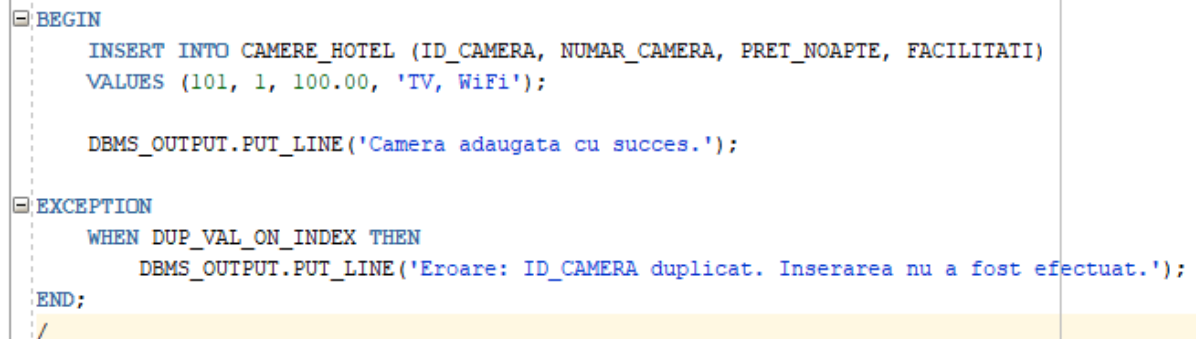
```
EXCEPTION
```

```
    WHEN DUP_VAL_ON_INDEX THEN
```

```
        DBMS_OUTPUT.PUT_LINE('Eroare: ID_CAMERA duplicat. Inserarea nu a fost efectuat.');
```

```
END;
```

```
/
```

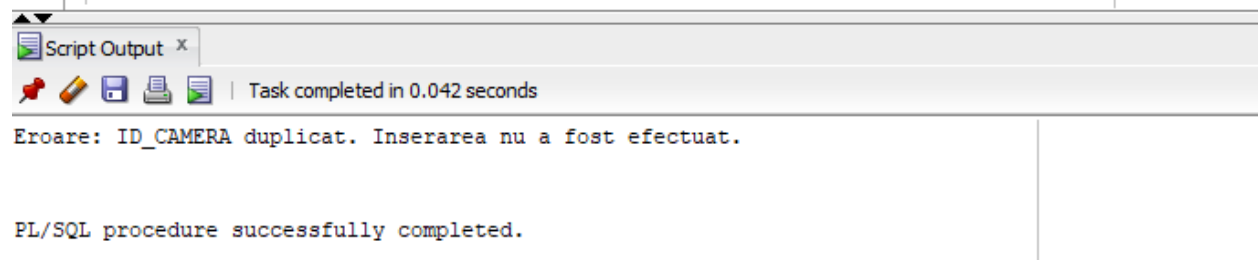


```
BEGIN
    INSERT INTO CAMERE_HOTEL (ID_CAMERA, NUMAR_CAMERA, PRET_NOAPTE, FACILITATI)
    VALUES (101, 1, 100.00, 'TV, WiFi');

    DBMS_OUTPUT.PUT_LINE('Camera adaugata cu succes.');
```

```
EXCEPTION
    WHEN DUP_VAL_ON_INDEX THEN
        DBMS_OUTPUT.PUT_LINE('Eroare: ID_CAMERA duplicat. Inserarea nu a fost efectuat.');
```

```
END;
/
```



Script Output x

Task completed in 0.042 seconds

Eroare: ID_CAMERA duplicat. Inserarea nu a fost efectuat.

PL/SQL procedure successfully completed.

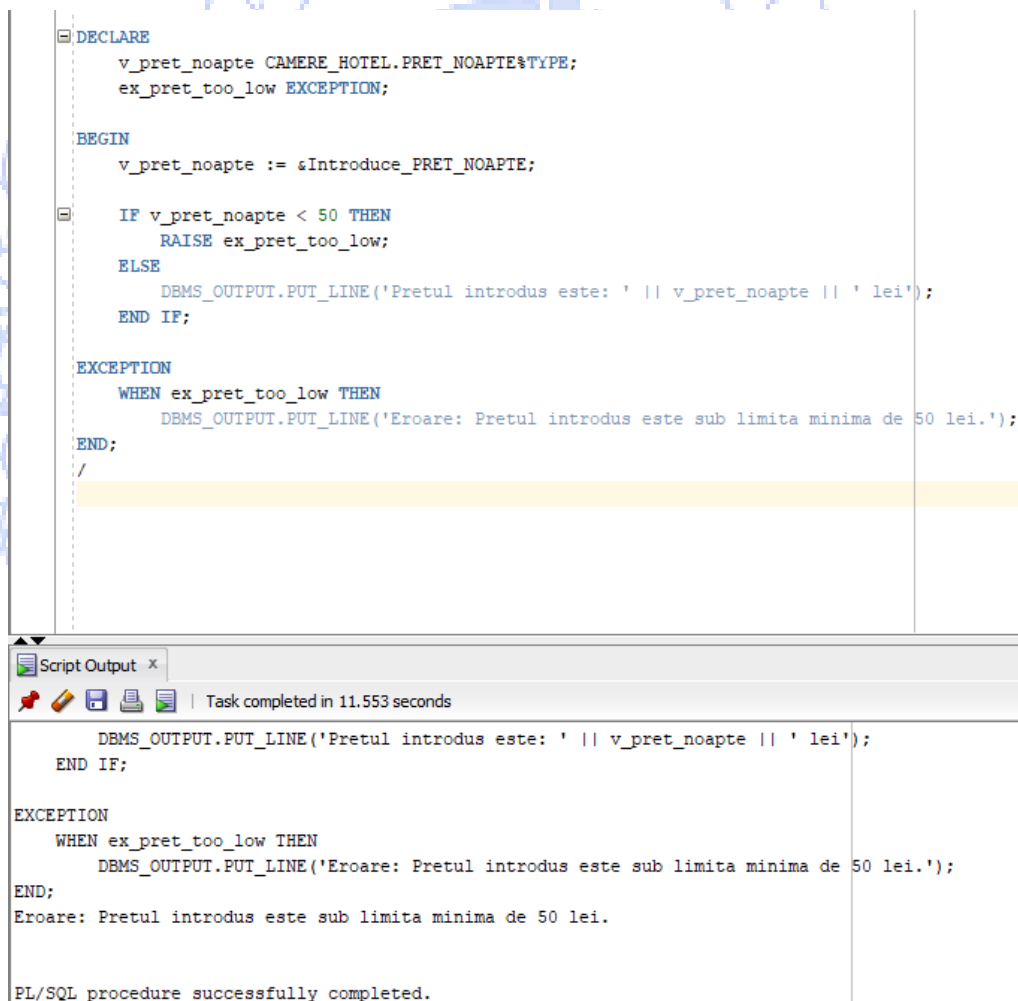
--Exceptie personalizata pentru preturile camerelor

```
DECLARE
```

```

v_pret_noapte CAMERE_HOTEL.PRET_NOAPTE%TYPE;
ex_pret_too_low EXCEPTION;
BEGIN
    v_pret_noapte := &Introduce_PRET_NOAPTE;
    IF v_pret_noapte < 50 THEN
        RAISE ex_pret_too_low;
    ELSE
        DBMS_OUTPUT.PUT_LINE('Pretul introdus este: ' || v_pret_noapte || ' lei');
    END IF;
EXCEPTION
    WHEN ex_pret_too_low THEN
        DBMS_OUTPUT.PUT_LINE('Eroare: Pretul introdus este sub limita minima de 50 lei.');
```

END;



```

-- DECLARE
v_pret_noapte CAMERE_HOTEL.PRET_NOAPTE%TYPE;
ex_pret_too_low EXCEPTION;

-- BEGIN
v_pret_noapte := &Introduce_PRET_NOAPTE;

-- IF v_pret_noapte < 50 THEN
RAISE ex_pret_too_low;
-- ELSE
DBMS_OUTPUT.PUT_LINE('Pretul introdus este: ' || v_pret_noapte || ' lei');
-- END IF;

-- EXCEPTION
-- WHEN ex_pret_too_low THEN
DBMS_OUTPUT.PUT_LINE('Eroare: Pretul introdus este sub limita minima de 50 lei.');
```

END;

/

Script Output x

Task completed in 11.553 seconds

```

DBMS_OUTPUT.PUT_LINE('Pretul introdus este: ' || v_pret_noapte || ' lei');
END IF;

EXCEPTION
    WHEN ex_pret_too_low THEN
        DBMS_OUTPUT.PUT_LINE('Eroare: Pretul introdus este sub limita minima de 50 lei.');
```

END;

Eroare: Pretul introdus este sub limita minima de 50 lei.

PL/SQL procedure successfully completed.

--Exceptie TOO_MANY_ROWS pentru rezervari multiple

```

DECLARE
    v_id_client ISTORIC_REZERVARI.ID_CLIENT%TYPE := &Introduce_ID_CLIENT;
    v_id_rezervare ISTORIC_REZERVARI.ID_REZERVARE%TYPE;

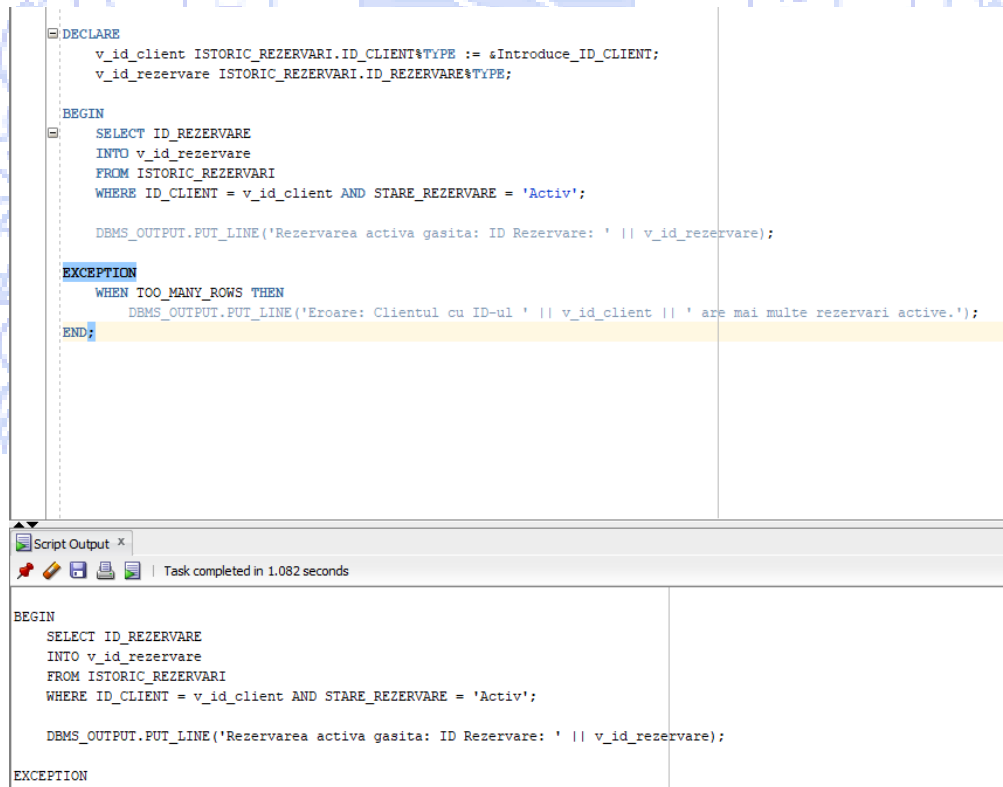
BEGIN
    SELECT ID_REZERVARE
    INTO v_id_rezervare
    FROM ISTORIC_REZERVARI
    WHERE ID_CLIENT = v_id_client AND STARE_REZERVARE = 'Activ';

    DBMS_OUTPUT.PUT_LINE('Rezervarea activa gasita: ID Rezervare: ' || v_id_rezervare);

EXCEPTION
    WHEN TOO_MANY_ROWS THEN

        DBMS_OUTPUT.PUT_LINE('Eroare: Clientul cu ID-ul ' || v_id_client || ' are mai multe
rezervari active.');
```

END;



```

DECLARE
    v_id_client ISTORIC_REZERVARI.ID_CLIENT%TYPE := &Introduce_ID_CLIENT;
    v_id_rezervare ISTORIC_REZERVARI.ID_REZERVARE%TYPE;

BEGIN
    SELECT ID_REZERVARE
    INTO v_id_rezervare
    FROM ISTORIC_REZERVARI
    WHERE ID_CLIENT = v_id_client AND STARE_REZERVARE = 'Activ';

    DBMS_OUTPUT.PUT_LINE('Rezervarea activa gasita: ID Rezervare: ' || v_id_rezervare);

EXCEPTION
    WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('Eroare: Clientul cu ID-ul ' || v_id_client || ' are mai multe rezervari active.');
```

END;

Script Output x

Task completed in 1.082 seconds

```

BEGIN
    SELECT ID_REZERVARE
    INTO v_id_rezervare
    FROM ISTORIC_REZERVARI
    WHERE ID_CLIENT = v_id_client AND STARE_REZERVARE = 'Activ';

    DBMS_OUTPUT.PUT_LINE('Rezervarea activa gasita: ID Rezervare: ' || v_id_rezervare);

EXCEPTION
```

Functii

Sa se determine suma totala a platilor efectuate de un client, identificat prin ID-ul sau. Daca nu exista plati, sa se returneze 0

--Calculare suma totala a platilor pentru un client

```
CREATE OR REPLACE FUNCTION suma_plati_client(p_id_client IN NUMBER) RETURN NUMBER IS
```

```
    v_suma_totala NUMBER;
```

```
BEGIN
```

```
    SELECT NVL(SUM(suma_plata), 0)
```

```
    INTO v_suma_totala
```

```
    FROM plati
```

```
    WHERE id_client = p_id_client;
```

```
    RETURN v_suma_totala;
```

```
END;
```

```
/
```

```
SELECT suma_plati_client(1) AS suma_totala FROM dual;
```

```
CREATE OR REPLACE FUNCTION suma_plati_client(p_id_client IN NUMBER) RETURN NUMBER IS
    v_suma_totala NUMBER;
BEGIN
    SELECT NVL(SUM(suma_plata), 0)
    INTO v_suma_totala
    FROM plati
    WHERE id_client = p_id_client;

    RETURN v_suma_totala;
END;
/
SELECT suma_plati_client(1) AS suma_totala FROM dual;
```

Script Output x

Task completed in 0.097 seconds

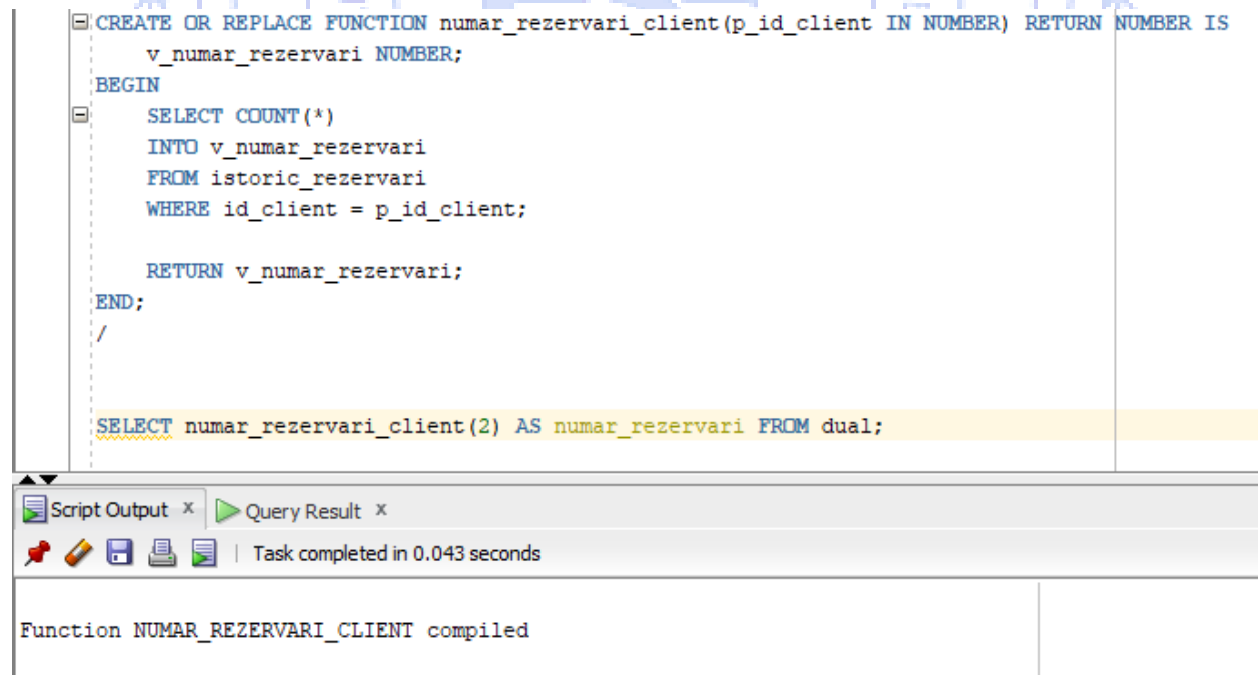
Function SUMA_PLATI_CLIENT compiled

Script Output x		Query Result x	
		SQL All Rows Fetched: 1 in 0.01 seconds	
		SUMA_TOTALA	
1			500

Sa se afiseze numarul total de rezervari efectuate de un client identificat prin ID-ul sau.

```
CREATE OR REPLACE FUNCTION numar_rezervari_client(p_id_client IN NUMBER) RETURN NUMBER IS
    v_numar_rezervari NUMBER;
BEGIN
    SELECT COUNT(*)
    INTO v_numar_rezervari
    FROM istoric_rezervari
    WHERE id_client = p_id_client;

    RETURN v_numar_rezervari;
END;
/
SELECT numar_rezervari_client(2) AS numar_rezervari FROM dual;
```



The screenshot shows the SQL Developer interface. The main editor contains the PL/SQL code for creating a function and executing it. The 'Script Output' pane at the bottom shows the message 'Function NUMAR_REZERVARI_CLIENT compiled'. The 'Query Result' pane shows the result of the query.

```
CREATE OR REPLACE FUNCTION numar_rezervari_client(p_id_client IN NUMBER) RETURN NUMBER IS
    v_numar_rezervari NUMBER;
BEGIN
    SELECT COUNT(*)
    INTO v_numar_rezervari
    FROM istoric_rezervari
    WHERE id_client = p_id_client;

    RETURN v_numar_rezervari;
END;
/
SELECT numar_rezervari_client(2) AS numar_rezervari FROM dual;
```

Script Output x Query Result x

Task completed in 0.043 seconds

Function NUMAR_REZERVARI_CLIENT compiled

NUMAR_REZERVARI
1

Sa se verifice daca o camera este disponibila intr-un anumit interval de timp, identificat prin data de check-in si check-out.

```

CREATE OR REPLACE FUNCTION camera_disponibila(
    p_id_camera IN NUMBER,
    p_data_check_in IN DATE,
    p_data_check_out IN DATE
) RETURN NUMBER IS
    v_count NUMBER;
BEGIN
    SELECT COUNT(*)
    INTO v_count
    FROM istoric_rezervari
    WHERE id_camera = p_id_camera
        AND stare_rezervare = 'Activ'
        AND (
            (p_data_check_in BETWEEN data_check_in AND data_check_out) OR
            (p_data_check_out BETWEEN data_check_in AND data_check_out) OR
            (p_data_check_in <= data_check_in AND p_data_check_out >= data_check_out)
        );
    IF v_count = 0 THEN
        RETURN 1; -- Camera este disponibilă
    ELSE
        RETURN 0; -- Camera nu este disponibilă
    END IF;
END;

SELECT camera_disponibila(101, TO_DATE('2022-01-10', 'YYYY-MM-DD'), TO_DATE('2023-01-15',
'YYYY-MM-DD')) AS disponibila FROM dual;

--deoarece SQq/PL NU FOLOSESTE tipuri de date Boolean vom folosi un identificatory de tip 1 si
0

```

```

CREATE OR REPLACE FUNCTION camera_disponibila(
    p_id_camera IN NUMBER,
    p_data_check_in IN DATE,
    p_data_check_out IN DATE
) RETURN NUMBER IS
    v_count NUMBER;
BEGIN
    SELECT COUNT(*)
    INTO v_count
    FROM istoric_rezervari
    WHERE id_camera = p_id_camera
        AND stare_rezervare = 'Activ'
        AND (
            (p_data_check_in BETWEEN data_check_in AND data_check_out) OR
            (p_data_check_out BETWEEN data_check_in AND data_check_out) OR
            (p_data_check_in <= data_check_in AND p_data_check_out >= data_check_out)
        );

    IF v_count = 0 THEN
        RETURN 1;
    ELSE
        RETURN 0;
    END IF;
END;
/

```

Script Output x

Task completed in 0.087 seconds

Function CAMERA_DISPONIBILA compiled

Script Output x Query Result x

SQL | All Rows Fetched

DISPONIBILA

1	1
---	---

Proceduri

Sa se afiseze toate rezervarile active, incluzand ID-ul rezervarii, clientul asociat, camera si perioadele de check-in si check-out.

--Afisarea rezervarilor active

```
CREATE OR REPLACE PROCEDURE afisare_rezervari_active IS
    CURSOR rezervari_cursor IS
        SELECT id_rezervare, id_client, id_camera, data_check_in, data_check_out
        FROM istoric_rezervari
        WHERE stare_rezervare = 'Activ';
BEGIN
    FOR rezervare_rec IN rezervari_cursor LOOP
        DBMS_OUTPUT.PUT_LINE('ID Rezervare: ' || rezervare_rec.id_rezervare ||
            ', ID Client: ' || rezervare_rec.id_client ||
            ', ID Camera: ' || rezervare_rec.id_camera ||
            ', Check-in: ' || TO_CHAR(rezervare_rec.data_check_in, 'DD-MON-
YYYY') ||
            ', Check-out: ' || TO_CHAR(rezervare_rec.data_check_out, 'DD-MON-
YYYY'));
    END LOOP;
END;
/

BEGIN
    afisare_rezervari_active;
END;
/
```

Worksheet Query Builder

```

CREATE OR REPLACE PROCEDURE afisare_rezervari_active IS
    CURSOR rezervari_cursor IS
        SELECT id_rezervare, id_client, id_camera, data_check_in, data_check_out
        FROM istoric_rezervari
        WHERE stare_rezervare = 'Activ';
BEGIN
    FOR rezervare_rec IN rezervari_cursor LOOP
        DBMS_OUTPUT.PUT_LINE('ID Rezervare: ' || rezervare_rec.id_rezervare ||
            ', ID Client: ' || rezervare_rec.id_client ||
            ', ID Camera: ' || rezervare_rec.id_camera ||
            ', Check-in: ' || TO_CHAR(rezervare_rec.data_check_in, 'DD-MON-YYYY') ||
            ', Check-out: ' || TO_CHAR(rezervare_rec.data_check_out, 'DD-MON-YYYY'));
    END LOOP;
END;
/

BEGIN
    afisare_rezervari_active;
END;
/

```

Script Output x

Task completed in 0.095 seconds

Procedure AFISARE_REZERVARI_ACTIVE compiled

```

ID Rezervare: 1, ID Client: 1, ID Camera: 101, Check-in: 15-MAR-2023, Check-out: 20-MAR-2023
ID Rezervare: 3, ID Client: 3, ID Camera: 103, Check-in: 01-MAY-2023, Check-out: 05-MAY-2023
ID Rezervare: 5, ID Client: 5, ID Camera: 105, Check-in: 10-JUL-2023, Check-out: 15-JUL-2023
ID Rezervare: 7, ID Client: 7, ID Camera: 107, Check-in: 01-SEP-2023, Check-out: 05-SEP-2023
ID Rezervare: 9, ID Client: 9, ID Camera: 109, Check-in: 01-NOV-2023, Check-out: 05-NOV-2023

```

PL/SQL procedure successfully completed.

Sa se mareasca pretul pe noapte al camerelor care au o facilitate specificata (ex. "WiFi") cu un anumit procent.

--Marim pretul camerelor cu unele facilitate

```

CREATE OR REPLACE PROCEDURE mareste_pret_camere(
    p_facilitate IN VARCHAR2,
    p_procent IN NUMBER
) IS
BEGIN
    UPDATE camere_hotel
    SET pret_noapte = pret_noapte * (1 + p_procent / 100)

```

```
WHERE facilitati LIKE '%' || p_facilitate || '%';
```

```
DBMS_OUTPUT.PUT_LINE('Preturile camerelor cu facilitatea "' || p_facilitate || '" au fost marite cu ' || p_procent || '%');
```

```
END;
```

```
/
```

```
EXEC mareste_pret_camere('WiFi', 10);
```

```
CREATE OR REPLACE PROCEDURE mareste_pret_camere(
  p_facilitate IN VARCHAR2,
  p_procent IN NUMBER
) IS
BEGIN
  UPDATE camere_hotel
  SET pret_noapte = pret_noapte * (1 + p_procent / 100)
  WHERE facilitati LIKE '%' || p_facilitate || '%';

  DBMS_OUTPUT.PUT_LINE('Preturile camerelor cu facilitatea "' || p_facilitate || '" au fost marite cu ' || p_procent || '%');
END;
/
EXEC mareste_pret_camere('WiFi', 10);
```

Script Output x

Task completed in 0.096 seconds

Procedure MARESTE_PRET_CAMERE compiled

Preturile camerelor cu facilitatea "WiFi" au fost marite cu 10%

PL/SQL procedure successfully completed.

Sa se calculeze si sa se afiseze valoarea totala a fiecarei rezervari dintr-un an specificat.

--Rezervarile dintr-un respective an

```
CREATE OR REPLACE PROCEDURE valoare_rezervari(p_an IN NUMBER) IS
  CURSOR rezervari_cursor IS
    SELECT ir.id_rezervare,
           ir.data_check_in,
           SUM(ch.pret_noapte * (ir.data_check_out - ir.data_check_in)) AS valoare
    FROM istoric_rezervari ir
    JOIN camere_hotel ch ON ir.id_camera = ch.id_camera
```

```

WHERE EXTRACT(YEAR FROM ir.data_check_in) = p_an

GROUP BY ir.id_rezervare, ir.data_check_in;

BEGIN

FOR rezervare IN rezervari_cursor LOOP

    DBMS_OUTPUT.PUT_LINE('Rezervarea cu ID ' || rezervare.id_rezervare ||

        ' din data ' || TO_CHAR(rezervare.data_check_in, 'DD-MON-YYYY')

||

        ' are valoarea: ' || rezervare.valoare || ' RON.');
```

END LOOP;

END;

/

BEGIN

valoare_rezervari(2023);

END;

```

CREATE OR REPLACE PROCEDURE valoare_rezervari(p_an IN NUMBER) IS
    CURSOR rezervari_cursor IS
        SELECT ir.id_rezervare,
            ir.data_check_in,
            SUM(ch.pret_noapte * (ir.data_check_out - ir.data_check_in)) AS valoare
        FROM istoric_rezervari ir
        JOIN camere_hotel ch ON ir.id_camera = ch.id_camera
        WHERE EXTRACT(YEAR FROM ir.data_check_in) = p_an
        GROUP BY ir.id_rezervare, ir.data_check_in;
BEGIN
    FOR rezervare IN rezervari_cursor LOOP
        DBMS_OUTPUT.PUT_LINE('Rezervarea cu ID ' || rezervare.id_rezervare ||
            ' din data ' || TO_CHAR(rezervare.data_check_in, 'DD-MON-YYYY') ||
            ' are valoarea: ' || rezervare.valoare || ' RON.');
```

END LOOP;

END;

/

BEGIN

valoare_rezervari(2023);

END;

Script Output x

Task completed in 0.024 seconds

```

Rezervarea cu ID 4 din data 01-JUN-2023 are valoarea: 387.2 RON.
Rezervarea cu ID 5 din data 10-JUL-2023 are valoarea: 1210 RON.
Rezervarea cu ID 6 din data 01-AUG-2023 are valoarea: 435.6 RON.
Rezervarea cu ID 7 din data 01-SEP-2023 are valoarea: 629.2 RON.
Rezervarea cu ID 8 din data 01-OCT-2023 are valoarea: 532.4 RON.
Rezervarea cu ID 9 din data 01-NOV-2023 are valoarea: 363 RON.
Rezervarea cu ID 10 din data 01-DEC-2023 are valoarea: 871.2 RON.
```

PL/SQL procedure successfully completed.